



## PATENT ABSTRACTS OF JAPAN

(11) Publication number: **07020778 A**(43) Date of publication of application: **24.01.95**

(51) Int. Cl. **G09C 1/00**  
**G06F 7/52**  
**G06F 7/72**

(21) Application number: **05164870**(22) Date of filing: **02.07.93**(71) Applicant: **FUJITSU LTD**

(72) Inventor: **TAKENAKA MASAHIKO**  
**TORII NAOYA**  
**HASEBE TAKAYUKI**  
**AKIYAMA RYOTA**

(54) **REMAINDER CALCULATING DEVICE, TABLE  
 GENERATING DEVICE, AND MULTIPLICATION  
 REMAINDER CALCULATING DEVICE**

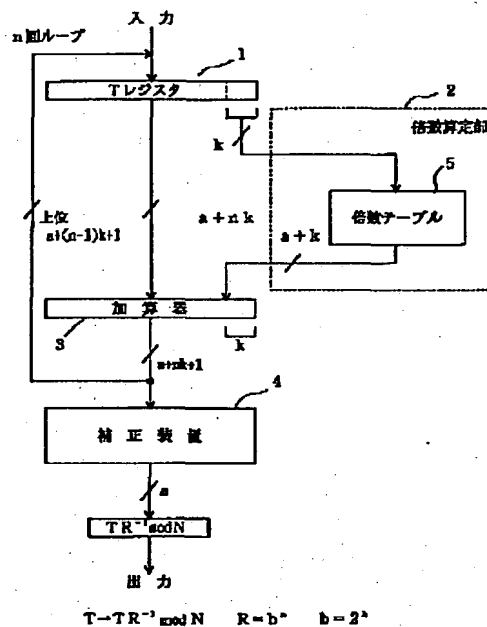
addition result of the adder 3.

COPYRIGHT: (C)1995,JPO

## (57) Abstract:

**PURPOSE:** To enable a high-speed processing by simplifying an arithmetic processing for remainder calculation.

**CONSTITUTION:** A multiple table 5 containing multiplies of a modulus N of a remainder corresponding to prescribed low-order bits of an input register 1 is provided and retrieved on the basis of prescribed low-order bits of a number to be subjected to the remainder calculation T in the input register 1 to obtain a multiple of the modulus N of the remainder corresponding to the prescribed low-order bits of T. An adder 3 adds the multiple of the modulus N of the remainder, obtained by the indexing in the multiple table 5, to the contents of the input register 1. The contents of the input register 1 are updated with the prescribed high-order bits of the addition result of the adder 3 each time the addition by the adder 3 is performed (n) times as prescribed. A correcting device 4 performs a correcting processing for (t) obtained as the



改定

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平7-20778

(43) 公開日 平成7年(1995) 1月24日

(51) Int. Cl. <sup>8</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 9 C 1/00		8837-5L		
G 0 6 F 7/52	A			
7/72				

審査請求 未請求 請求項の数10 O L (全 16 頁)

(21) 出願番号 特願平5-164870

(22) 出願日 平成5年(1993) 7月2日

(71) 出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中1015番地

(72) 発明者 武仲 正彦

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72) 発明者 鳥居 直哉

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72) 発明者 長谷部 高行

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(74) 代理人 弁理士 遠山 勉 (外1名)

最終頁に続く

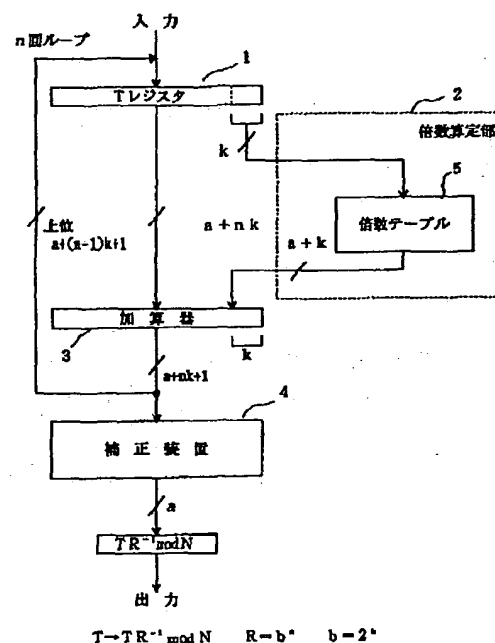
(54) 【発明の名称】 剰余計算装置、テーブル作成装置および乗算剰余計算装置

(57) 【要約】

【目的】 剰余計算の演算処理を簡略化して、高速処理を実現する。

【構成】 入力レジスタ1の下位所定ビットに対応して剰余の法Nの倍数が格納された倍数テーブル5を有し、入力レジスタ1の被剰余数Tの下位所定ビットに基づいて倍数テーブル5を検索することにより、前記Tの下位所定ビットに対応する剰余の法Nの倍数を求める。加算器3は、倍数テーブル5の索引により求められる剰余の法Nの倍数と入力レジスタ1の内容とを加算する。加算器3の所定回数nの加算について、加算器3の加算結果の上位所定ビットで入力レジスタ1の内容を更新する。補正装置4は、加算器3の加算結果として得られるtに対して補正処理を施す。

剰余計算装置の原理図



## 【特許請求の範囲】

【請求項 1】 入力される被剰余数を保持する入力レジスタ (1) と、

前記入力レジスタ (1) に格納される被剰余数の下位所定ビットに基づいて、予め演算データを格納したテーブルを用いるテーブル検索により、前記被剰余数の下位所定ビットに対応する剰余の法の倍数を求めるための倍数算定手段 (2) と、

前記倍数算定手段 (2) で求められる剰余の法の倍数と前記入力レジスタ (1) の内容とを加算するとともに、所定回数の加算についてその加算結果の上位所定ビットで前記入力レジスタ (1) の内容を更新するための加算手段 (3) と、

前記加算手段 (3) の加算結果に対して剰余結果の補正処理を施すための補正手段 (4) とを具備することを特徴とする剰余計算装置。

【請求項 2】 倍数算定手段 (2) は、被剰余数の下位所定ビットに対応させて剰余の法の倍数を格納した倍数テーブル (5) を有し、この倍数テーブル (5) を前記被剰余数の下位所定ビットで検索して、それに対応する前記剰余の法の倍数を直接求める手段であることを特徴とする請求項 1 に記載の剰余計算装置。

【請求項 3】 倍数算定手段 (2) は、被剰余数の下位所定ビットに対応させて倍数情報を格納した倍数情報テーブル (4 1) を前記被剰余数の下位所定ビットで検索して、それに対応する倍数情報を求める倍数情報算定手段 (4 2) と、剰余の法を格納する剰余の法レジスタ (1 2) と、前記倍数情報算定手段 (4 2) で得られる倍数情報と前記剰余の法レジスタ (1 2) から得られる剰余の法とを乗算する乗算手段 (1 4) とを有し、前記倍数算定手段 (1 0) により前記被剰余数の下位所定ビットに対応する前記剰余の法の倍数を求める手段であることを特徴とする請求項 1 に記載の剰余計算装置。

【請求項 4】 請求項 2 の剰余計算装置に用いる倍数テーブルを記憶装置上に作成するテーブル作成装置において、剰余の法を繰り返し加算するための繰り返し加算手段 (6 2) と、前記繰り返し加算手段 (6 2) で求められる各加算結果の値の下位  $b$  ビットの 2 の補数を求めるための補数算定手段 (6 4) と、前記補数算定手段 (6 4) で得られる各補数値に対応するテーブルアドレスに前記各加算結果をそれぞれ格納するためのテーブル書き込み手段とを具備することを特徴とするテーブル作成装置。

【請求項 5】 被乗数の下位から所定ビット毎の値と乗数を乗算し、その乗算結果に剰余を加算する分割乗算装置 (7 5) と、この分割乗算装置 (7 5) から出力される被剰余数を保持する被剰余数レジスタ (1 1) と、前記被剰余数レジスタ (1 1) に格納される被剰余数の下位所定ビットに基づいて、予め演算データを格納した

テーブルを用いるテーブル検索により、前記被剰余数の下位所定ビットに対応する剰余の法の倍数を求めるための倍数算定手段と、

前記倍数算定手段で求められる剰余の法の倍数と前記入力レジスタの内容とを加算するとともに、所定回数の加算についてその加算結果を前記剰余として前記分割乗算装置の前記乗算結果への加算に供するための加算手段 (1 6) と、

前記加算手段 (1 6) の加算結果に対して剰余結果の補正処理を施すための補正手段 (2 2) とを有することを特徴とする乗算剰余計算装置。

【請求項 6】 倍数算定手段は、被剰余数の下位所定ビットに対応させて剰余の法の倍数を格納した倍数テーブル (5 1) を有し、この倍数テーブル (5 1) を前記被剰余数の下位所定ビットで検索して、それに対応する前記剰余の法の倍数を直接求める手段であることを特徴とする請求項 5 に記載の乗算剰余計算装置。

【請求項 7】 倍数算定手段は、被剰余数の下位所定ビットに対応させて倍数情報を格納した倍数情報テーブル (4 1) を前記被剰余数の下位所定ビットで検索して、それに対応する倍数情報を求めるための倍数情報算定手段と、剰余の法を格納する剰余の法レジスタ (1 2) と、前記倍数情報算定手段で得られる倍数情報と前記剰余の法レジスタ (1 2) から得られる剰余の法とを乗算するための乗算手段 (1 4) とを有し、この倍数算定手段により前記被剰余数の下位所定ビットに対応する前記剰余の法の倍数を求める手段であることを特徴とする請求項 5 に記載の乗算剰余計算装置。

【請求項 8】 整数  $T$  および  $N$  について、被剰余数  $T$  に対する剰余の法  $N$  における剰余  $T \bmod N = \text{REDC}(T * (R^2 \bmod N))$  を求めるため  $TR^{-1} \bmod N = \text{REDC}(T)$  を算出する剰余計算装置において、

入力される被剰余数  $T$  を保持する入力レジスタ (1 1) と、

前記入力レジスタ (1 1) に格納される被剰余数  $T$  の下位所定ビットに基づいて、予め演算データを格納したテーブルを用いるテーブル検索により、前記被剰余数の下位所定ビットに対応する剰余の法  $N$  の倍数  $m' N$  (但し、 $m' = (T \bmod b) N' \bmod b$ 、ここで  $N'$  および  $R$  は整数であり、 $R = b^n$ 、 $b = 2^k$  とする) を求めるための倍数算定手段と、

前記倍数算定手段で求められる剰余の法  $N$  の倍数  $m' N$  と前記入力レジスタの内容とを加算するとともに、所定回数の加算についてその加算結果の上位所定ビットで前記入力レジスタの内容を更新することにより  $t = (T + mN) / R$  を出力する加算器 (1 6) と、

前記加算器 (1 6) の加算結果  $t$  に対して  $t < N$  の場合はそのまま  $t = \text{REDC}(T)$ 、 $t \geq N$  の場合は  $t - N = \text{REDC}(T)$  として補正処理を施し  $TR^{-1} \bmod N$  を得る補正装置 (2 2) とを有することを特徴とする剰余

計算装置。

【請求項 9】 倍数算定手段 (52) は、被剰余数  $T$  の下位所定ビットに対応させて剰余の法  $N$  の倍数  $m' \cdot N$  を格納した倍数テーブル (51) を有し、この倍数テーブル (51) を前記被剰余数  $T$  の下位所定ビットで検索して、それに対応する前記剰余の法の倍数  $m' \cdot N$  を直接求める手段であることを特徴とする請求項 8 に記載の剰余計算装置。

【請求項 10】 倍数算定手段は、被剰余数  $T$  の下位所定ビットに対応させて倍数情報  $m'$  を格納した倍数情報テーブル (41) を前記被剰余数  $T$  の下位所定ビットで検索して、それに対応する倍数情報  $m'$  を求める倍数情報算定手段 (42) と、剰余の法  $N$  を格納する剰余の法レジスタ (12) と、前記倍数情報算定手段 (42) で得られる倍数情報  $m'$  と前記剰余の法レジスタ (12) から得られる剰余の法  $N$  とを乗算する乗算器 (14) とを有し、この倍数算定手段により前記被剰余数の下位所定ビットに対応する前記剰余の法  $N$  の倍数  $m' \cdot N$  を求める手段であることを特徴とする請求項 8 に記載の剰余計算装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、例えば公開鍵暗号系における RSA 暗号処理における剰余演算等に好適な剰余計算装置に係り、特に剰余計算の一手法であるモンゴメリのアルゴリズム (Modulo Multiplication Without Trial Division, Peter L. Montgomery, Mathematics of Computation, Volume 44, Number 170, April 1985 pp. 519~528 参照) を用いて、高速に剰余計算を行う剰余計算装置および乗算剰余計算装置ならびに剰余計算に使用するための倍数テーブルを作成するテーブル作成装置に関する。

【0002】

【従来の技術】 近年のコンピュータネットワークの発達により、データベースの検索や電子メール、電子ニュースなどの電子化された情報をネットワークを経由して送受する機会が急速に増加してきている。さらに、これらを利用して、オンラインショッピングなどのサービスも提供されつつある。しかし、それにとまって、ネットワーク上の電子化されたデータを盗聴する、改ざんする、他人になりすましてサービスを受けるなどの問題も指摘されている。特に無線を利用したネットワークにおいては、傍受が容易なためにこれらを防止する対策が望まれている。

【0003】 これらの問題に対して暗号技術 (encryption technology) を応用した暗号化電子メールや利用者認証システムが提案され、種々のネットワークにも導入されつつある。この意味でコンピュータネットワークにおいては暗号化は必須の技術であるといえる。このような暗号技術の中の 1 つにデジタル署名すなわち認証に適

した公開鍵暗号方式 (public key cryptosystem) があるが、暗号化/復号に大量の処理が必要なため高速化が望まれており、様々な高速化アルゴリズムも発表されている。

【0004】 暗号化方式は、大別すると秘密鍵暗号系と公開鍵暗号系の 2 つに分類することができる。秘密鍵暗号系は、送信者と受信者が同じ暗号鍵を持つことにより暗号通信を行う方式である。すなわち、秘密暗号系では、あるメッセージを秘密の暗号鍵に基づいて暗号化して相手に送り、受け手はこの暗号鍵を用いて暗号を復号しもとのメッセージに戻して情報を入手する。

【0005】 公開鍵暗号系は、送信者は公開されている受信者の公開鍵でメッセージを暗号化して送信し、受信者は自分の秘密鍵でその暗号化メッセージを復号することで通信を行う方式である。すなわち、公開鍵暗号系では、公開鍵は暗号化のための鍵、秘密鍵は公開鍵により暗号化された暗号を復号するための鍵であり、公開鍵で暗号化した暗号は秘密鍵でのみ復号することができる。

【0006】 秘密鍵暗号系では、個人が秘密に保管しなければならない鍵の数が通信相手の数だけ必要であり、必要な総鍵数は  $n$  人のネットワークの場合  $n(n-1)/2$  個である。また、はじめて通信する相手に対しては、何らかの方法で秘密鍵の配送が必要であるという欠点がある。この問題を避けるために、大規模なネットワークでは鍵管理センタを設置し、センタとの間の秘密鍵のみを保管し、暗号通信を行う場合はセンタから送信相手との秘密鍵を得る方法が用いられる。この場合秘密鍵の総数は  $n$  となる。

【0007】 一方、公開鍵暗号系では、個人が秘密に保管する鍵は自分の秘密鍵のみであり、必要な総秘密件数も  $n$  人のネットワークの場合  $n$  個である。また、はじめて通信する相手に対しては、公開鍵の配送を行えばよく、鍵管理センタを設置して、ユーザの公開鍵を  $n$  個公開簿に登録し、センタから送信相手の公開鍵を得る方法が用いられる。この場合、センタは公開鍵の改ざんを防ぐだけで、秘密に保管する必要がない。但し、公開鍵方式は秘密鍵方式に比べて鍵のビット数が大きいので保管に要するファイルサイズは大きくなる。

【0008】 また、認証の場合、秘密暗号系では、例えば送信するメッセージを秘密鍵で圧縮変換し、送信文に付加して送り、受信側では同様に圧縮変換して比較する方式がとられている。しかし、送受信が同じ鍵であるため受信者は認証データを偽造することができる。これに対して、公開鍵暗号系では、秘密鍵で暗号化することができるのは本人だけであるという特徴を利用する。送信者はメッセージを圧縮変換して秘密鍵で暗号化し、送信文に付加して送り、受信者は送信者の公開鍵で付加されたデータを復号し、同様に圧縮変換したものと比較する方式がとられている。この場合は受信者は不正がで

【0009】このように、認証系では公開鍵暗号系の技術は必要不可欠であるといえる。しかし、公開鍵暗号系には、暗号化/復号に大量の処理が必要であるという大きな欠点があるため、一般には処理の速い秘密暗号系をメッセージの暗号化に、公開鍵暗号系は認証用というように組み合わせて用いられる場合が多い。公開暗号系の中で、現在最も有力なものが1977年にリヴェスト(Rivest)、シャミア(Shamir)およびエイドルマン(Adlman)の3人によって発明されたRSA暗号である。RSA暗号の基本原理は次のようなものである。

【0010】〈RSAの基本アルゴリズム〉暗号鍵

(e, N) と対応する復号鍵 (d, N) で、e と N は公開鍵であり、d は秘密鍵である。平文を M、暗号文を C とすると、暗号化 E と複合化 D のアルゴリズムは次のようにあらわされる。

$$\begin{aligned} C &= E(M) = M^e \bmod N \\ M &= D(C) = C^d \bmod N \end{aligned}$$

但し、 $d \cdot e = 1 \bmod \text{LCM}\{(p-1), (q-1)\}$

$N = p \cdot q$

LCM: 最小公倍数[lowest common multiple]; p, q は大きな素数である。

【0011】通常、e, d, N, Mなどは512ビット程度の大きな整数が用いられるので、高速指数計算法を使用しても1回のRSA演算で平均770回程度の乗算と剰余演算を行わなければならない。特に剰余演算は、近似法や剰余テーブル方式、モンゴメリのアルゴリズム等、多くの高速化手法が提案されている。このような、RSA暗号に代表される公開鍵暗号系の多くで利用される、べき乗剰余アルゴリズムを高速に処理するためには、1回あたりの剰余アルゴリズムの高速化が要求される。

【0012】この剰余演算の高速化の実現の一方法であるモンゴメリのアルゴリズムについて説明する。

〈モンゴメリのアルゴリズム〉モンゴメリのアルゴリズムは、剰余の法 N ( $N > 1$ ) と、剰余の法 N と互いに素である基数 R ( $R > N$ ) を用いると、被剰余数 T から  $TR^{-1} \bmod N$  の計算が基数 R による除算のみで行えることを利用して、N による除算を用いることなく剰余計算を行うアルゴリズムである。ここで、N, N', R,  $R^{-1}$  および T は整数であり、被剰余数 T は  $0 \leq T < RN$ 、 $R^{-1}$  は剰余の法 N の上での基数 R の逆数であり、 $RR^{-1} \bmod N = 1$  ( $0 \leq R^{-1} < N$ ,  $0 \leq N' < R$ ) の関係を満たす。

【0013】さらにこの基数 R に 2 のべき乗数を使用した場合、基数 R による除算をシフト操作に置き換えることができるため、 $T \rightarrow TR^{-1} \bmod N$  の計算の高速処理が可能となる。次にアルゴリズム 1 として、 $T \rightarrow TR^{-1} \bmod N$  のアルゴリズム REDC (T) を示す。但し、アル

ゴリズム 1 において  $(T + mN) / R$  は必ず割り切れる。

【0014】〈アルゴリズム 1〉 $T \rightarrow TR^{-1} \bmod N$  のアルゴリズム REDC (T) は次のようにあらわされる。

```
05 m = (T mod R) N' mod R
   t = (T + mN) / R
   if t < N then return t else return t - N
   すなわち、
   REDC (T) = t      (t < N)
10 = t - N      (t ≥ N)
   である。
```

【0015】1回の REDC では、剰余  $T \bmod N$  ではなく  $TR^{-1} \bmod N$  が求められるだけである。そこで、剰余  $T \bmod N$  を求めるためには、次に示すように REDC

```
15 (T) と予め求めておいた  $R^2 \bmod N$  との積で、再び REDC
   DCを行えばよい。
   REDC (REDC (T) * ( $R^2 \bmod N$ ))
   = ( $TR^{-1} \bmod N$ ) ( $R^2 \bmod N$ )  $R^{-1} \bmod N$ 
   =  $TR^{-1} * R^2 * R^{-1} \bmod N$ 
20 =  $T \bmod N$ 
```

このようにして、剰余  $T \bmod N$  を求めることができる。

【0016】〈REDCの多重精度計算への拡張〉次に、剰余の法 N あるいは基数 R が多倍長すなわち多重精度の場合について、REDCのアルゴリズムを拡張する。剰余の法 N や基数 R が多重精度の場合、REDCの

```
25 ( $T \bmod R$ ) N' や mN の計算は、多重精度 × 多重精度
   の処理となり汎用の計算機では非常に大きな処理量と処理
   時間が必要となる。そこで、この部分を多重精度 × 単
   精度の処理で行えるように拡張したアルゴリズム 2 を示
30 す。
```

【0017】〈アルゴリズム 2〉REDCを多重精度へ

拡張したアルゴリズムは次に示すようになる。被剰余数 T が b 進数で  $T = (T_{2n-1} T_{2n-2} \dots T_0)_b$ 、 $R = b^n$ 、 $b = 2^k$  とあらわされる場合、次に示す  $i = 0 \sim n-1$

```
35 の繰り返し処理により  $TR^{-1} \bmod N$  を単倍長と同様にして
   求めることができる。
   【0018】
   for i = 0 to n - 1
       m' =  $T_i N' \bmod b$ 
       T =  $T + m' N$ 
40   t =  $T / b$ 
   next
   if t < N then return t else return t - N
```

このようにして得られる  $TR^{-1} \bmod N$  と、上述したよう

```
45 に予め求めておいた  $R^2 \bmod N$  との積で再び REDC を行
   うことにより求める剰余  $T \bmod N$  を求めることができる。
```

【0019】

【発明が解決しようとする課題】上述したモンゴメリのアルゴリズムは、基数 R を 2 のべき乗数にすることによ

り、処理時間のかかる割り算を高速なシフト処理に置き換えることが可能であるため剰余計算を高速で行うことができる。しかし、このモンゴメリのアルゴリズムを用いた演算を高速で行おうとすると、多重精度の処理は、多重精度×単精度の乗算処理が必要となるため、大量の処理が必要となる。

【0020】また、べき乗剰余などの演算をこのアルゴリズムを用いて行う場合、剰余演算が何度も繰り返されることになるため、剰余演算の処理回数の多い部分をできるだけ単純化することで全体の計算量を削減し、処理速度の向上を図る必要がある。また、モンゴメリ法特有の $N'$  および $R^2 \bmod N$ 等の事前に計算しておかなければならないパラメータの計算が前処理として必要である。特に、 $N'$  は、拡張ユークリッドの互除法を使用するため、 $R^2 \bmod N$ よりも多くの処理時間を必要とする。

【0021】したがって、本発明は、剰余計算に際し多重の繰り返し演算を必要とする倍数演算部分に着目して剰余計算の演算処理を簡略化し、高速処理を実現し得る剰余計算装置および乗算剰余計算装置を提供することを目的としている。本発明の他の目的は、本発明による剰余計算装置および乗算剰余計算装置に用いるための倍数テーブルを容易に作成することのできるテーブル作成装置を提供することにある。

#### 【0022】

【課題を解決するための手段】本発明は、上記目的を達成するために、多重の繰り返し演算を必要とする倍数演算部分にテーブルを用い、演算結果をテーブル検索により得るようにして、演算処理を簡略化して高速化することにより、剰余計算を高速処理し得る次のような構成の剰余計算装置とした。

【0023】図1は、本発明の基本となる剰余計算装置の原理を示している。図1に示す剰余計算装置は、入力レジスタ1、倍数算定部2、加算器3および補正装置4を有している。入力レジスタ1は入力される被剰余数 $T$ を保持する。倍数算定部2は、入力レジスタ1の下位所定ビットに対応して剰余の法 $N$ の倍数が格納された倍数テーブル5を有し、入力レジスタ1の被剰余数 $T$ の下位所定ビットに基づいて倍数テーブル5を検索することにより、前記 $T$ の下位所定ビットに対応する剰余の法 $N$ の倍数を求める。加算器3は、倍数算定部2で求められる剰余の法 $N$ の倍数と入力レジスタ1の内容とを加算する。この加算器3の所定回数 $n$ の加算について、加算器3の加算結果の上位所定ビットで入力レジスタ1の内容を更新する。補正装置4は、加算器3の加算結果として得られる $t$ に対して補正処理を施す。

【0024】このような剰余計算装置は、 $n$ 回の繰り返し演算の必要な倍数算定部2に倍数テーブル5を用いて高速に倍数を求めることができるので、剰余計算を高速に処理することができる。図1の剰余計算装置においては、具体的には、先に述べたモンゴメリのアルゴリズム

における被剰余数 $T$ が入力レジスタ1に格納され、倍数算定部2において剰余の法 $N$ の倍数 $m' \cdot N$ （但し、 $m' = (T \bmod b) \cdot N' \bmod b$ 、ここで $N'$  および $R$ は整数であり、 $R = b^n$ 、 $b = 2^k$ とする）が格納された倍数テーブル5を検索して、前記被剰余数 $T$ の下位 $k$ ビットに対応する剰余の法 $N$ の倍数 $m' \cdot N$ が求められる。この倍数 $m' \cdot N$ と前記被剰余数 $T$ が加算され、その加算結果により入力レジスタ1の内容が更新される。このような処理が $n$ 回繰り返され、加算器3の出力に $t = (T + m \cdot N) / R$ が得られる。この加算器3の出力 $t$ は、補正装置4に与えられ、 $t < N$ の場合はそのまま $t = \text{REDC}(T)$ 、 $t \geq N$ の場合は $t - N = \text{REDC}(T)$ として、 $TR^{-1} \bmod N$ が得られる。

【0025】上述の倍数テーブル5を作成するテーブル作成装置は、繰り返し加算部、補数算定部およびテーブル書き込み部を有する。繰り返し加算部は剰余の法 $N$ を繰り返し加算する。補数算定部は前記繰り返し加算部で求められる各加算結果の値の下位 $b$ ビットの2の補数を求める。テーブル書き込み部は、テーブルの前記補数算定部で得られる各補数値に対応するテーブルアドレスに前記各加算結果をそれぞれ格納する。このテーブル作成装置は、例えば同様の機能をソフトウェアにより実現するコンピュータにより構成することもできる。

【0026】このようなテーブル作成装置により、計算処理に多くの時間を必要とする $N'$ を求めることなく記憶装置上に倍数テーブルを作成することができる。また、上述の倍数テーブル5は、多量のデータにより構成されることになり、倍数テーブル5を構成する記憶装置に多くの記憶容量を必要とするので、倍数テーブル5を用いる倍数算定部2に代えて、被剰余数 $T$ の下位所定ビットに対応させて倍数情報 $m'$ を格納する倍数情報テーブルを用い、この倍数情報テーブルを前記被剰余数 $T$ の下位所定ビットで検索して、それに対応する倍数情報 $m'$ を求める倍数情報算定部と、剰余の法 $N$ を格納する剰余の法レジスタと、前記倍数情報算定部で得られる倍数情報 $m'$ と前記剰余の法レジスタから得られる剰余の法 $N$ とを乗算する乗算器とを有する倍数算定部を設ける構成としてもよい。この場合、前記倍数算定部により前記被剰余数の下位所定ビットに対応する前記剰余の法 $N$ の倍数 $m' \cdot N$ を求める。この場合、計算速度は、倍数テーブル5を用いる場合よりも遅くなるが、倍数情報テーブルは倍数テーブル5に比して少ない記憶容量で済む。

【0027】さらに、乗算剰余計算装置は、上述した剰余計算装置と同様の構成に加えて、被乗数に対し所定ビット毎に乗数との乗算を行い行う分割乗算装置を設け、この分割乗算装置の出力を上述した剰余計算装置の入力レジスタに与えるとともに、前記剰余計算装置の加算器の出力を前記分割乗算装置の乗算結果に加算する構成として、乗算剰余の計算を高速に行う。

#### 【0028】

【作用】本発明による剰余計算装置では、剰余計算において、ループによる多数回の繰り返し演算が必要な倍数算定部2に、入力レジスタ1の保持値の下位所定ビットに基づくテーブル検索を用いて高速に処理することにより、高速剰余計算が行われる。特に、倍数算定部2に倍数テーブル5を用いて、倍数算定部2の機能をすべてテーブル検索により処理すれば、処理速度が顕著に向上する。

【0029】本発明によるテーブル作成装置では、倍数算定部2に用いる倍数テーブル5を作成するのに、剰余の法を繰り返し加算し、加算結果の下位所定ビットの2の補数を求め、その2の補数値に対応するアドレスに前記加算結果を格納することにより、煩雑な計算を要するパラメータ $N'$ を求めることなく倍数テーブル5を作成することができる。

【0030】本発明による乗算剰余計算装置では、剰余計算における繰り返し処理の部分にテーブルを用いて、効率よく剰余計算処理を高速化し、乗算剰余計算を処理速度を向上させる。

#### 【0031】

##### 【実施例】

〈実施例1〉本発明に係る剰余計算装置の第1の実施例の説明に先立ち、本発明における剰余計算処理の原理を詳細に説明する。上述したモンゴメリのアルゴリズムを利用して、ハードウェアにより剰余計算装置を構成すると、一般的には図8のようになると考えられる。図8の構成は、先に述べた多重精度のアルゴリズムをそのままハードウェアで実現したものである。

【0032】図8の剰余計算装置は、先に述べた通り、 $R=b^n$ 、 $b=2^k$ として、 $T \rightarrow TR^{-1} \bmod N$ の処理を行うものとし、Tレジスタ11、Nレジスタ12、 $N'$ レジスタ13、第1の乗算器14、第2の乗算器15、第1の加算器16、第1のレジスタ17、 $-N$ レジスタ18、第2の加算器19、第2のレジスタ20およびセレクタ21を具備する。

【0033】第1のレジスタ17、 $-N$ レジスタ18、第2の加算器19、第2のレジスタ20およびセレクタ21は、補正装置22を構成する。 $N'$ レジスタ13および第2の乗算器15からなる部分は倍数情報算定部31を構成し、この倍数情報算定部31、Nレジスタ12および第1の乗算器14からなる部分は倍数算定部32を構成する。

【0034】ここで、剰余の法Nをaビット、入力値を $a+nk$ ビットと仮定して、上述の剰余計算装置について説明する。Tレジスタ11は、図1における入力レジスタ1に相当し、入力される被剰余数Tを保持する。Nレジスタ12および $N'$ レジスタ13は、共にaビットのレジスタであり、与えられる剰余の法Nおよび先に述べた $N'$ をそれぞれ保持する。第2の乗算器15は、Tレジスタ11の保持値の下位kビットと $N'$ レジスタ1

3の下位kビットとを乗算する。第1の乗算器14は、Nレジスタ12の保持値と第2の乗算器15の乗算結果の下位kビットとを乗算する。

【0035】第1の加算器16は、Tレジスタ11の保持値( $a+nk$ ビット)と第1の乗算器14の乗算結果( $a+k$ ビット)とを加算する。第1の加算器16の加算結果( $a+nk+1$ ビット)の上位 $a+(n-1)k+1$ ビットはTレジスタ11の内容を更新してn回のループ処理を行うのに用いられる。第1のレジスタ17は、第1の加算器16の加算結果の下位kビットを超える部分の下位 $a+1$ ビットを保持し、 $-N$ レジスタ18は、予め与えられる剰余の法Nに基づく $-N$ の値( $a+1$ ビット)を保持する。第2の加算器19は、第1のレジスタ17の保持値と $-N$ レジスタ18の保持値とを加算して、第2のレジスタ20に保持させる。セレクタ21は、第2のレジスタ20の $a+1$ ビットの保持値の最上位の1ビット(符号ビットに相当する)の値を条件として動作し、該最上位ビットが“1”の場合は第1のレジスタ17の保持値を出力し、“0”の場合は第2のレジスタ20の保持値を出力する。

【0036】すなわち、剰余計算を行う場合、前処理として事前に計算して求めておいたN、 $N'$ および $-N$ を、Nレジスタ、 $N'$ レジスタおよび $-N$ レジスタにそれぞれ設定する。入力被剰余数Tは、まずTレジスタ11に与えられる。このTレジスタ11の保持値の下位kビットと $N'$ レジスタ13の保持値の下位kビットとを第2の乗算器15に入力する。第2の乗算器15の乗算結果の下位kビットとNレジスタ12の保持値を第1の乗算器14に入力し、その乗算結果とTレジスタ11の保持値とを第1の加算器16に入力する。Tレジスタ11と第1の加算器16との間のループ処理の回数がn未満の場合は、第1の加算器16の加算結果の上位 $a+(n-1)k+1$ ビットを再びTレジスタ11に入力する。ループ処理の回数がnに達した場合は、第1の加算器16の加算結果の下位 $k+1$ ビット目から下位 $a+k+2$ ビット目までの $a+1$ ビットを取り出し、補正装置22の第1のレジスタ17に入力する。

【0037】補正装置22では、入力を一旦第1のレジスタ17に保持し、その保持値と $-N$ レジスタ18の保持値とを第2の加算器19に入力し、加算結果を第2のレジスタ20に与える。第2のレジスタ20の上位1ビットの値をセレクタ21の動作条件として、“1”の場合は第1のレジスタ17の保持値が、“0”の場合は第2のレジスタ20の保持値がセレクタ21で選択され、結果として出力される。

【0038】ここで、補正装置22における処理は、モンゴメリのアルゴリズムで最後に計算値tがN以上の場合にはNを引く部分の処理である。この図8のような構成の場合、倍数算定部32の部分は何度も同じ処理を繰り返されることになる。さらにこの構成をべき乗剰余演

算に用いた場合はこれらの処理全体が繰り返されるため、倍数算定部 32 の部分の処理回数は非常に多くなる。

【0039】一方、倍数算定部 32 の処理自体は、乗算という処理量の大きな演算が中心であり、さらに計算内容が  $N$  という多重精度の定数と  $k$  ビットの変数との積であるため処理回数の多さとあいまって、長い処理時間が必要となる。そこで、本発明の第 1 の実施例では、図 8 における倍数情報算定部 31 の部分を  $k$  ビット  $\times 2^k$  の大きさの倍数情報テーブルとして持っておくことにより、無駄な計算を省き、処理を高速化する。

【0040】本発明に係る剰余計算装置の第 1 の実施例を説明する。なお、以下の各実施例においては、説明の便宜のためビット数を  $a=512$ 、 $k=8$ 、 $n=64$  として説明する。図 2 は、倍数情報テーブル方式を用いた剰余計算装置の構成を示している。図 2 において、図 8 と共通の部分には同符号を付してその詳細な説明を省略する。図 2 の剰余計算装置は、図 8 と同様の、Tレジスタ 11、Nレジスタ 12、第 1 の乗算器 14、第 1 の加算器 16 および補正装置 22 を具備する。補正装置 22 は、第 1 のレジスタ 17、 $-N$  レジスタ 18、第 2 の加算器 19、第 2 のレジスタ 20 およびセクタ 21 を有する。

【0041】さらに、図 2 の剰余計算装置においては、図 8 における倍数情報算定部 31 に代えて、倍数情報テーブル 41 を備えた倍数情報算定部 42 を設けている。倍数情報テーブル 41 は、被剰余数  $T$  の下位所定ビットに対応して倍数情報  $m'$  を格納している。Tレジスタ 11 は、1024 ビットの入力レジスタであり、この Tレジスタ 11 には、被剰余数  $T$  および第 1 の加算器 16 の出力ループが再び入力される。倍数情報テーブル 41 は、幅 8 ビット奥行き 256 のテーブルで、Tレジスタ 11 の下位 8 ビットで索引すると 8 ビットの倍数情報  $m'$  を出力する。Nレジスタ 12 は、剰余の法  $N$  を格納する 512 ビットのレジスタである。

【0042】第 1 の乗算器 14 は、8 ビット  $\times$  512 ビットの乗算器であり、倍数情報テーブル 41 の索引結果と Nレジスタ 12 の保持値との乗算を行う。第 1 の加算器 16 は、1025 ビットの加算器であり、Tレジスタ 11 の保持値と第 1 の乗算器 14 の乗算結果との加算を行う。補正装置 22 は、図 8 の補正装置 22 と同じもので剰余計算結果の補正を行う。

【0043】倍数情報算定部 42 の倍数情報テーブル 41 は、図 3 に示すように予め  $N'$  の下位 8 ビットを加算器により 255 回順次加算し、結果の下位 8 ビットを加算回数をアドレスとしてテーブル化したものである。被剰余数  $T$  が Tレジスタ 11 に入力されると、その下位 8 ビットで倍数情報テーブル 41 の索引を行い、その索引結果 ( $=m'$ ) と Nレジスタ 12 の保持値  $N$  を第 1 の乗算器 14 で乗算し、その積と Tレジスタ 11 の保持値と

を第 1 の加算器 16 で加算する。ループの回数  $n$  が 64 未満の場合は、第 1 の加算器 16 の加算結果の上位 1018 ビットを、再び Tレジスタ 11 に LSB (least significant bit) 側からつめて入力し、処理ループを形成する。ループの回数  $n$  が 64 回目の場合は、第 1 の加算器 16 の加算結果の下位 9 ビット目から 513 ビットを取り出して、補正装置 22 の第 1 のレジスタ 17 に入力する。補正処理は図 8 の場合と同様である。

【0044】このように、図 2 の剰余計算装置とすれば、倍数情報の算定を倍数情報算定部 42 でテーブル索引により行うので、倍数情報の算定を高速に行うことができ、剰余計算の高速化を実現することができる。なお、上述のように、図 8 における倍数情報算定部 31 の部分をテーブル化するよりも、図 8 における倍数算定部 32 の部分をテーブル化したほうが、当然高速化されるが、テーブルのサイズが大きくなるため、記憶装置に大きなテーブルをとる余裕がない場合に倍数情報算定部 31 の部分のテーブル化を行うことにより、小規模の装置構成でも高速化を実現することができる。

【0045】次に、図 8 における倍数算定部 32 の部分をテーブル化する場合の実施例を説明する。

〈実施例 2〉本発明の第 2 の実施例では、図 8 における倍数算定部 32 の部分を  $a+k$  ビット  $\times 2^k$  の大きさの倍数テーブルとして持っておくことにより、無駄な計算を省き、処理を高速化する。以下、本発明の第 2 の実施例に係る剰余計算装置を、図 2 または図 8 と同様の部分には同符号を付して示す図 4 を参照して説明する。

【0046】図 4 に示す倍数テーブル方式を用いた剰余計算装置は、図 2 と同様の、Tレジスタ 11、第 1 の加算器 16 および補正装置 22 を具備する。さらに、図 4 の剰余計算装置においては、図 8 における倍数算定部 32 に代えて、倍数テーブル 51 を備えた倍数算定部 52 を設ける。倍数算定部 52 の倍数テーブル 51 は、被剰余数  $T$  の下位所定ビットに対応させて剰余の法  $N$  の倍数  $m' \cdot N$  を格納している。すなわち、倍数テーブル 51 は、図 2 の倍数情報テーブル 41、Nレジスタ 12 および第 1 の乗算器 14 からなる部分の処理を一括してテーブル化したものであり、Tレジスタ 11 の保持値の下位 8 ビットで検索すると 520 ビットの倍数算定結果が出力される。この倍数テーブル 51 は、例えば後述する第 3 の実施例のような倍数テーブル作成方法に従って、予め幅 520 ビット奥行き 256 のテーブルとして作成する。

【0047】第 1 の実施例の場合と同様に被剰余数  $T$  が Tレジスタ 11 に入力されると、Tレジスタ 11 の下位 8 ビットで倍数算定部 52 の倍数テーブル 51 を索引し、その結果と Tレジスタ 11 の値を第 1 の加算器 16 で加算する。第 1 の加算器 16 の加算後の処理、つまり Tレジスタ 11 と第 1 の加算器 16 との間の加算ループ処理以後の処理は上述した第 1 の実施例と同様に行われ

05

10

15

20

25

30

35

40

45

50



る。

【0048】このように、図4の剰余計算装置とすれば、倍数算定を倍数算定部52で、場数テーブル51のテーブル索引により行うので、倍数の算定を一層高速に行うことができる。

〈実施例3〉本発明の第3の実施例は、上述した第2の実施例による図4に示した剰余計算装置で用いるための倍数テーブルを作成するためのテーブル作成装置であり、倍数テーブルを容易に作成することができる。

【0049】本発明の第3の実施例のテーブル作成装置は、モンゴメリのアルゴリズムの剰余の法Nの倍数を入力値Tに加えると、下位kビットが0となるという性質を利用し、パラメータN'を直接使用することなく倍数テーブルの作成を行う。図5にテーブル作成装置の構成および作成される倍数テーブルの構成を模式的に示している。

【0050】テーブル作成装置は、Nレジスタ61、加算器62、加算結果レジスタ63および補数演算部64を有する。Nレジスタ61は、剰余の法Nを保持し、このNレジスタ61の保持値を加算器62で繰り返し加算する。この加算器62の加算結果は加算結果レジスタ63に保持され、補数演算部64は、加算結果レジスタ63の保持値の下位所定ビットの2の補数を求める。このような計算の結果、逐次、補数演算部64で求められた値をテーブルのアドレスとして、加算結果レジスタ63に保持された値を書き込んで、倍数テーブル51とする。

【0051】ここでは、説明の便宜のため、剰余の法Nを10110011<sub>2</sub>とし、索引を4ビットで行うものとする。図5において、まず、0をテーブルの0番目すなわちアドレス0にセットする。次に0にNレジスタ61の剰余の法Nを加え、加算結果を10110011<sub>2</sub>として、加算結果レジスタ63に保持する。この加算結果の下位4ビットを取り出し、補数演算部64で2の補数をとると1101<sub>2</sub>になるので、テーブルのアドレス1101<sub>2</sub>に10110011<sub>2</sub>をセットする。このようにして、順次、テーブルの求められた2の補数値のアドレスに加算結果をセットしてゆく。

【0052】例えば、図5に示された状態では、剰余の法Nを6回加算した値10000110010<sub>2</sub>に、Nを加えて10011100101<sub>2</sub>を得て、下位4ビットの2の補数1011<sub>2</sub>をとる。そして、テーブルのアドレス1011<sub>2</sub>に10011100101<sub>2</sub>をセットしている。このような操作を16回繰り返すことにより、倍数テーブル51を作成することができる。

【0053】この実施例によるテーブル作成装置は、煩雑な計算を要するパラメータN'を直接使用することなく、モンゴメリのアルゴリズムの剰余の法Nの倍数を入力値Tに加えると、下位kビットが0となるという性質を利用して、倍数テーブルを容易に作成することができ

る。なお、このテーブル作成装置は、単純な繰り返し加算処理、2の補数を求める処理およびテーブルの書き込み処理のみで済むので、図5に対応するハードウェア構成のみならず、ワークステーション、パーソナルコンピュータ等を用いたソフトウェア処理で容易に実現することができる。

【0054】〈実施例4〉本発明の第4の実施例は、図2に示したように倍数情報テーブルを用いた剰余計算装置の構成を、分割乗算処理を行う分割乗算装置と組み合わせ構成した乗算剰余計算装置の実施例である。以下、本発明の第4の実施例に係る乗算剰余計算装置を、図2と同様の部分には同符号を付して示す図6を参照して説明する。

【0055】図6に示す乗算剰余計算装置は、 $R = b^n$ 、 $b = 2^k$ として、 $AB \rightarrow ABR^{-1} \bmod N$ の処理を行うものとし、図2と同様の、Tレジスタ11、Nレジスタ12、第1の乗算器14、第1の加算器16、補正装置22および倍数情報テーブル41を具備する。さらに、図6の乗算剰余計算装置においては、Aレジスタ71、Bレジスタ72、第3の乗算器73および第3の加算器74を設けている。

【0056】Aレジスタ71、Bレジスタ72、第3の乗算器73および第3の加算器74は分割乗算装置75を構成しており、Tレジスタ11、Nレジスタ12、第1の乗算器14、第1の加算器16および倍数情報テーブル41は剰余演算装置76を構成している。Aレジスタ71は与えられる乗数Aを保持し、Bレジスタ72は与えられる被乗数Bを保持する。乗算器73は、Bレジスタ72に保持される被乗数をkビット毎に下位から順次取り出して、Aレジスタ71の乗数Aに乗算する。加算器74は、n回のループ処理の間、剰余演算装置76の加算器16の演算結果を乗算器73の乗算結果に加算する。

【0057】この場合、図2の場合とは異なり、剰余演算装置76の加算器16の加算結果はそのまま加算器74に入力されてn回の処理ループを形成する。ここで、図6の乗算剰余計算装置は、図2におけるTレジスタ11と処理ループの部分に、512ビット幅のAレジスタ71およびBレジスタ72、520ビット幅の乗算器73、ならびに521ビット幅の加算器74を具備する分割乗算装置75を組み込んだ倍数情報テーブル式の乗算剰余計算装置として、具体的なビット数の例を示して説明する。図2の剰余計算装置をそのまま乗算剰余計算装置に使用する場合、乗算装置を設けて単にその乗算結果をTレジスタ11に代入するようにして実現することができるが、本実施例では、分割乗算装置と組み合わせることにより、剰余演算装置のTレジスタ11、加算器16およびループ処理部分のビット幅を削減している。

【0058】乗算剰余を行う2数すなわち乗数および被乗数が、AおよびBのレジスタ71および72にそれぞ

れ入力されると、Aレジスタ71の保持値とBレジスタ72の保持値の下位側から8ビットずつの値が乗算器73で乗算される。この乗算器73の乗算結果と前回の処理ループの計算値（初回は0とする）が加算器74で加算されてTレジスタ11に入力される。ここから加算器16までは図2の実施例の場合と同様の処理が行われる。ループの回数nが64未満の場合、加算器16の加算結果の上位514ビットをループによりフィードバックして加算器3に入力し、次のAレジスタ71とBレジスタ72との分割剰余結果に加算する。ループの回数nが64の場合、加算器16の加算結果の下位9ビット目から513ビットを取り出し、補正装置22に入力する。このようにすることにより、512ビットの $ABR^{-1} \bmod N$ が補正装置22から出力される。

【0059】（実施例5）本発明の第5の実施例は、図4に示したように倍数テーブルを用いた剰余計算装置の構成を、分割乗算処理を行う分割乗算装置と組み合わせて構成した乗算剰余計算装置の実施例である。以下、本発明の第5の実施例に係る乗算剰余計算装置を、図4または図6と同様の部分には同符号を付して示す図7を参照して説明する。

【0060】図7に示す乗算剰余計算装置は、図4と同様の、Tレジスタ11、第1の加算器16、補正装置22および倍数テーブル51を具備する。さらに、図7の乗算剰余計算装置においては、図6と同様の、Aレジスタ71、Bレジスタ72、第3の乗算器73および第3の加算器74を設けている。Aレジスタ71、Bレジスタ72、第3の乗算器73および第3の加算器74は分割乗算装置75を構成しており、Tレジスタ11、第1の加算器16および倍数テーブル51は剰余演算装置81を構成している。

【0061】図6の場合と同様に、Aレジスタ71は与えられる乗数Aを保持し、Bレジスタ72は与えられる被乗数Bを保持する。乗算器73は、Bレジスタ72に保持される被乗数をkビット毎に下位から順次取り出して、Aレジスタ71の乗数Aに乗算する。加算器74は、n回のループ処理の間、剰余演算装置81の加算器16の演算結果を乗算器73の乗算結果に加算する。加算器16の加算結果はそのまま加算器74に入力されてn回の処理ループを形成する。

【0062】なお、本発明の剰余計算装置および乗算剰余計算装置はハードウェアに限らず同様の機能構成の少なくとも一部をソフトウェアで実現することもでき、そのような場合にも処理の高速化を達成することができる。

#### 【0063】

【発明の効果】以上説明したように、本発明によれば、モンゴメリ法による剰余計算処理における倍数算定の乗算処理をテーブル索引で行うため、処理を簡略化ことができ、高速に剰余演算を行うことを可能とする剰余

計算装置および同様に高速に乗算剰余演算を行うことを可能とする乗算剰余計算装置を提供することができる。

【0064】例えば、ソフトウェアで本発明の剰余計算装置または乗算剰余計算装置を実現した場合、剰余計算部分を比較すれば、テーブルを用いない場合の処理時間を100とすると、倍数情報テーブル方式が98、倍数テーブル方式が31となり、その効果は明らかである。また、本発明によれば、倍数テーブル方式に用いる倍数テーブルを作成するのに、剰余の法を繰り返し加算し、加算結果の下位所定ビットの2の補数を求め、その2の補数値に対応するアドレスに前記加算結果を格納することにより、煩雑な計算を要するパラメータN'を求めることなくテーブルを作成することが可能なテーブル作成装置を提供することができる。

#### 【図面の簡単な説明】

【図1】本発明に係る剰余計算装置の構成を示す原理図である。

【図2】本発明の第1の実施例に係る剰余計算装置の構成を示すブロック図である。

【図3】図2の剰余計算装置で用いる倍数情報テーブルを説明するための模式図である。

【図4】本発明の第2の実施例に係る剰余計算装置の構成を示すブロック図である。

【図5】図4の剰余計算装置で用いる倍数テーブルを作成するための本発明の第3の実施例に係るテーブル作成装置の原理構成を概略的に示す図である。

【図6】本発明の第4の実施例に係る乗算剰余計算装置の構成を示すブロック図である。

【図7】本発明の第5の実施例に係る乗算剰余計算装置の構成を示すブロック図である。

【図8】モンゴメリのアルゴリズムを単純にハードウェア化した場合の一般的に考えられる剰余計算装置の構成を示すブロック図である。

#### 【符号の説明】

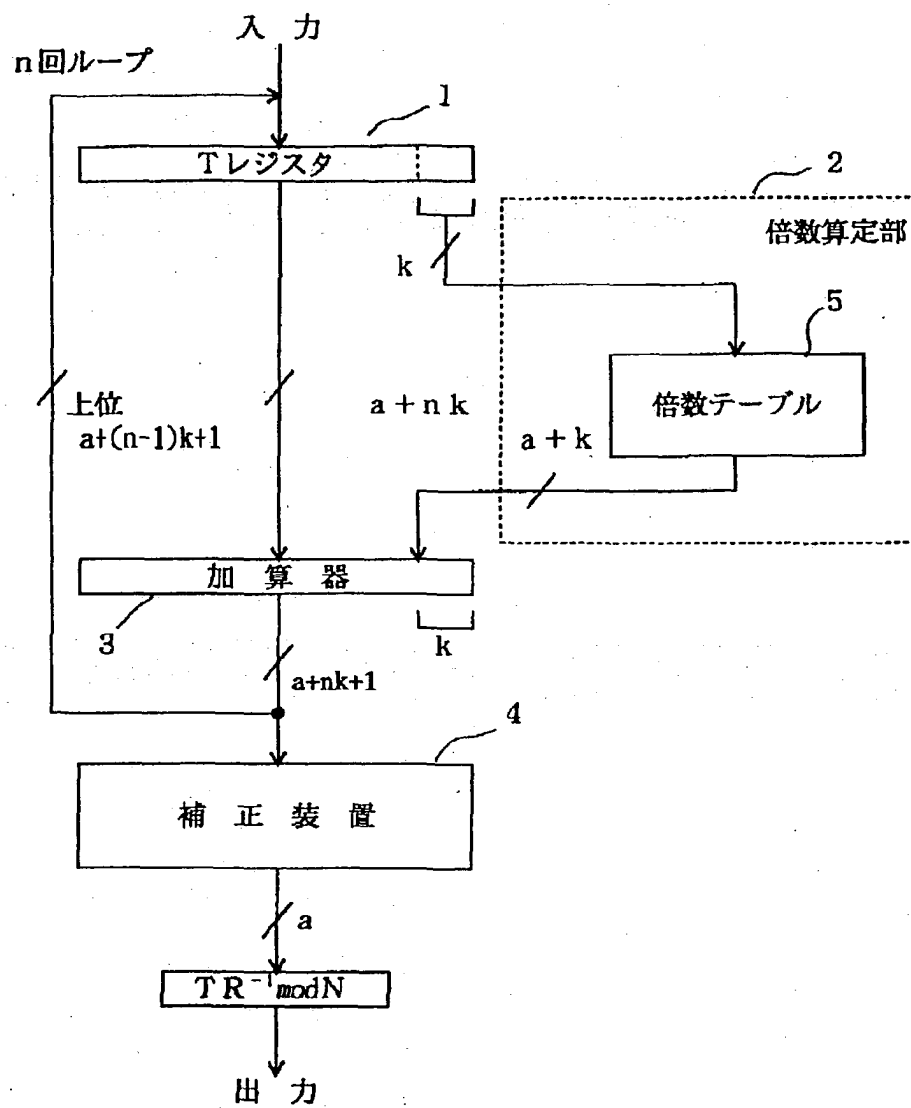
- 1, 11...Tレジスタ
- 2, 52...倍数算定部
- 3, 16, 19, 62, 74...加算器
- 4, 22...補正装置
- 5, 51...倍数テーブル
- 12, 61...Nレジスタ
- 14, 73...乗算器
- 17, 20...レジスタ
- 18...Nレジスタ
- 21...セクタ
- 41...倍数情報テーブル
- 42...倍数情報算定部
- 63...加算結果レジスタ
- 64...補数演算部
- 71...Aレジスタ
- 72...Bレジスタ

75...分割乗算装置

76, 81...剰余演算装置

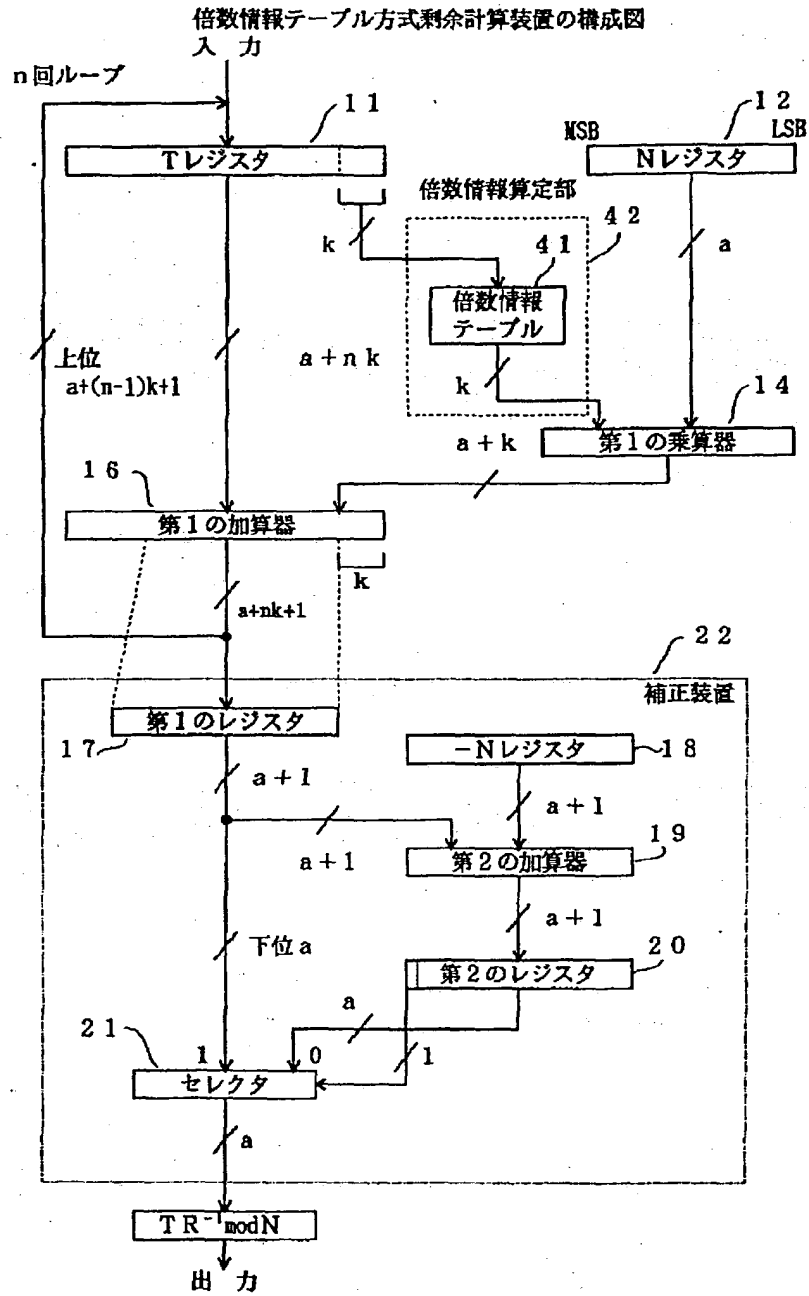
【図1】

剰余計算装置の原理図



$$T \rightarrow TR^{-1} \bmod N \quad R = b^n \quad b = 2^k$$

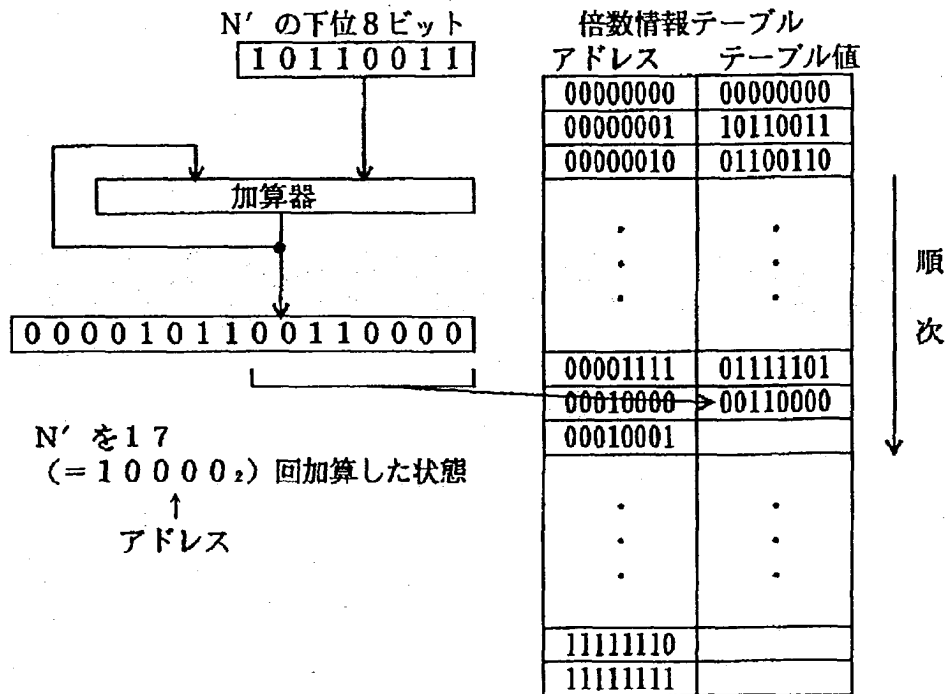
【図2】



$$T \rightarrow TR^{-1} \bmod N \quad R = b^n \quad b = 2^k$$

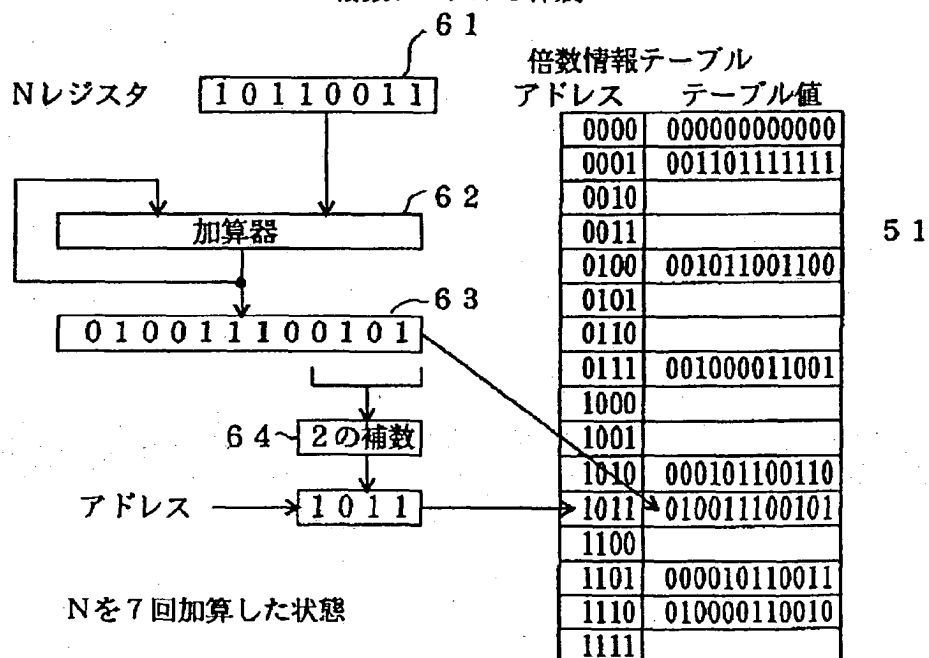
【図3】

## 倍数情報テーブルの作成

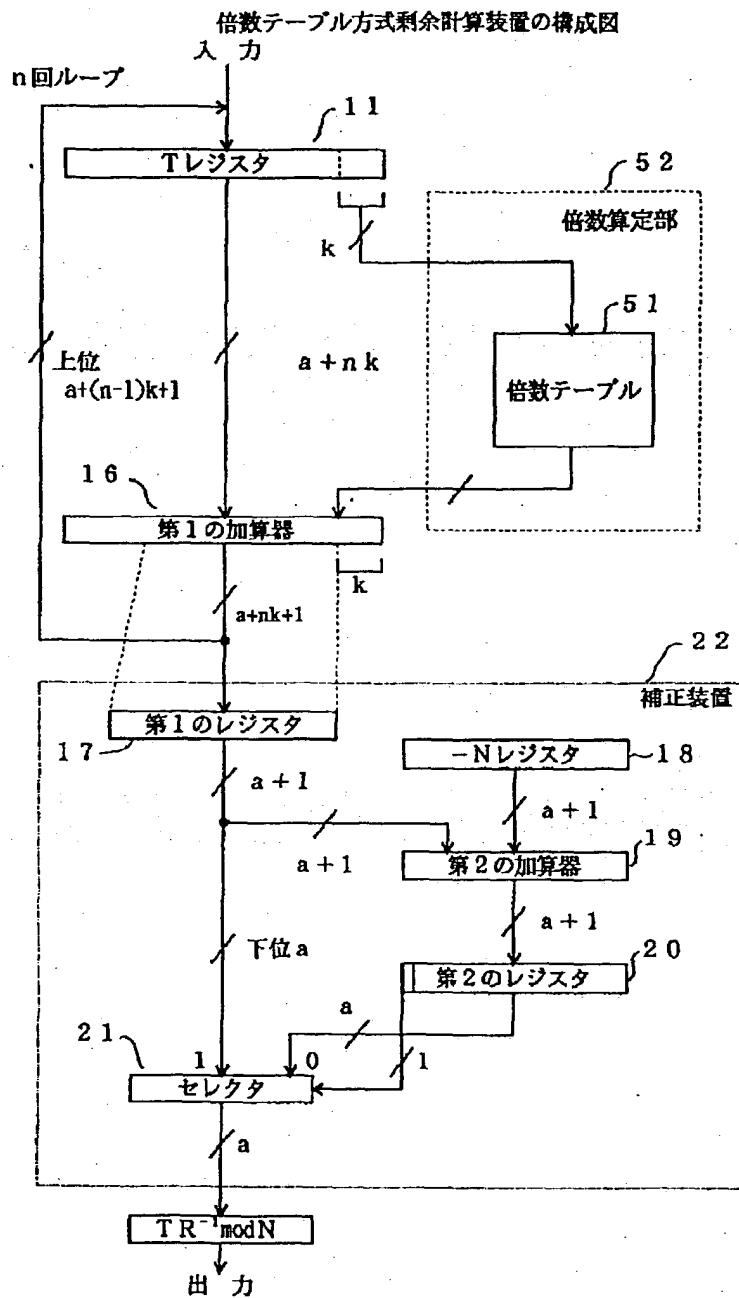


【図5】

## 倍数テーブルの作成



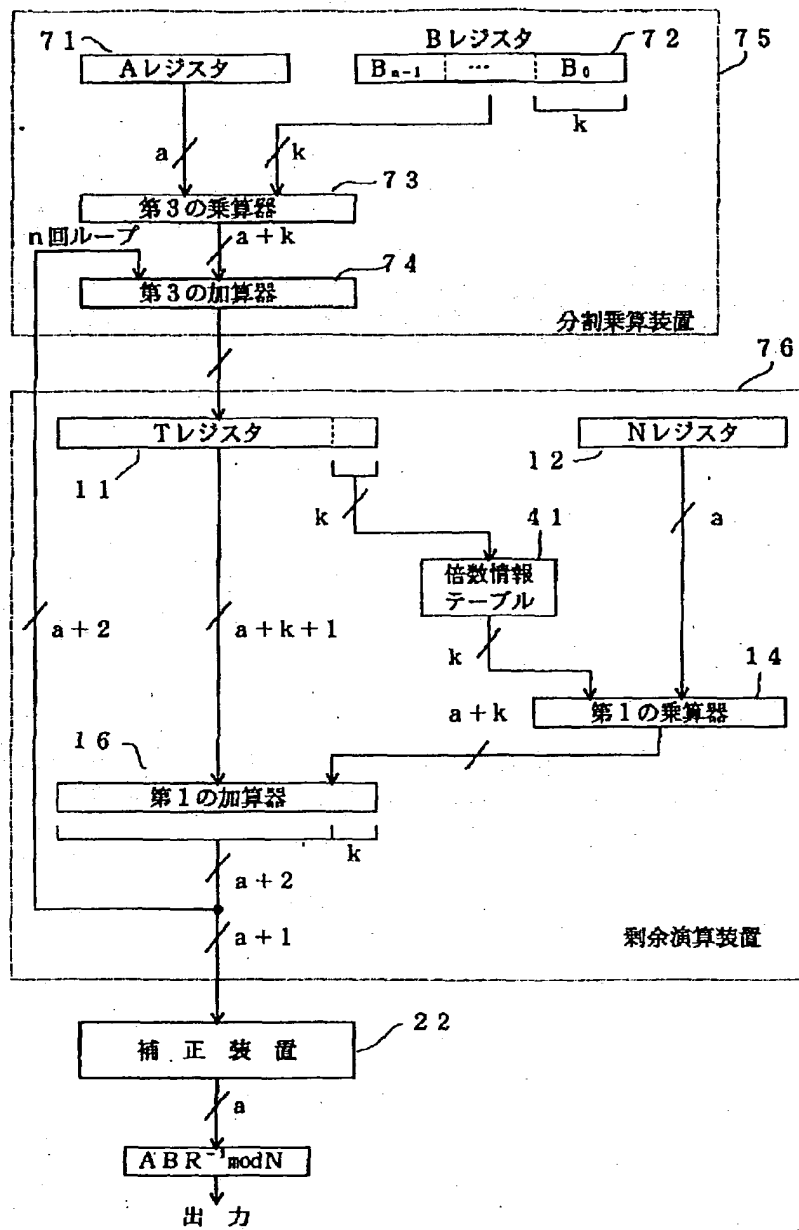
【図4】



$$T \rightarrow TR^{-1} \bmod N \quad R = b^a \quad b = 2^k$$

【図6】

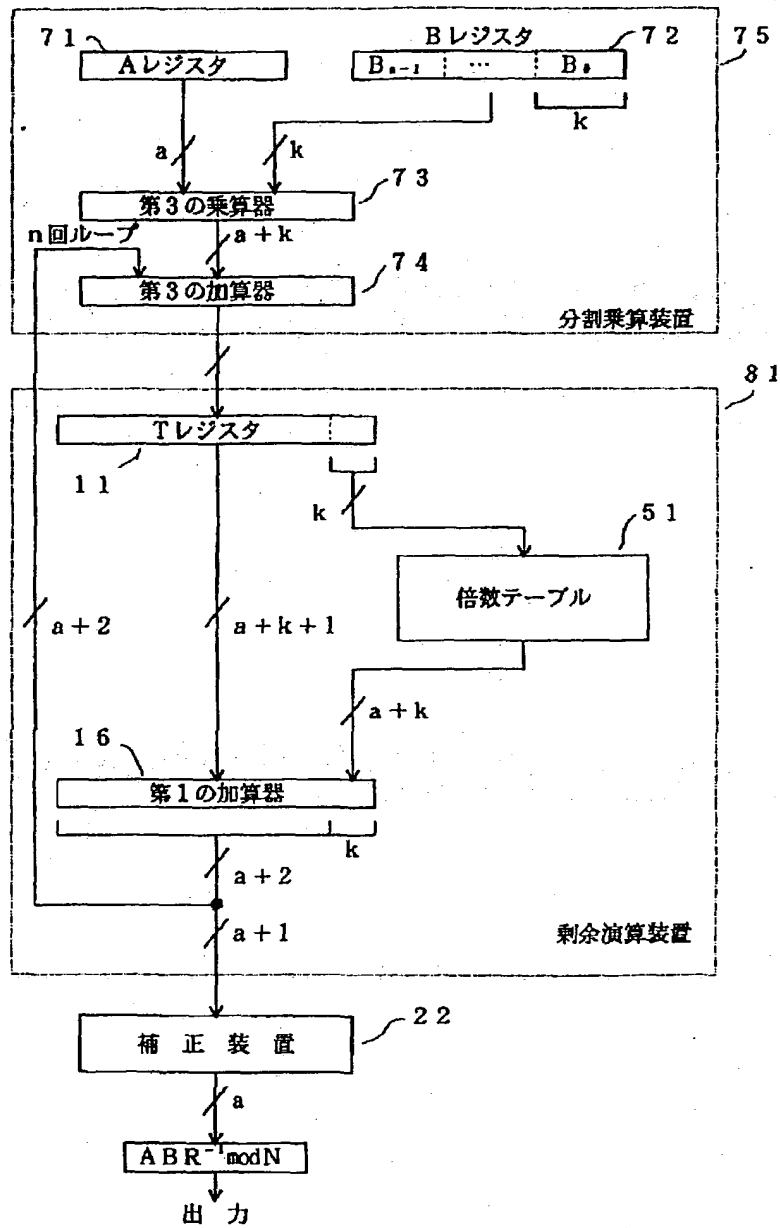
倍数情報テーブル方式乗算剰余計算装置の構成図



$$AB \rightarrow ABR^{-1} \bmod N \quad R = b^a \quad b = 2^k$$

【図7】

倍数テーブル方式乗算剰余計算装置の構成図

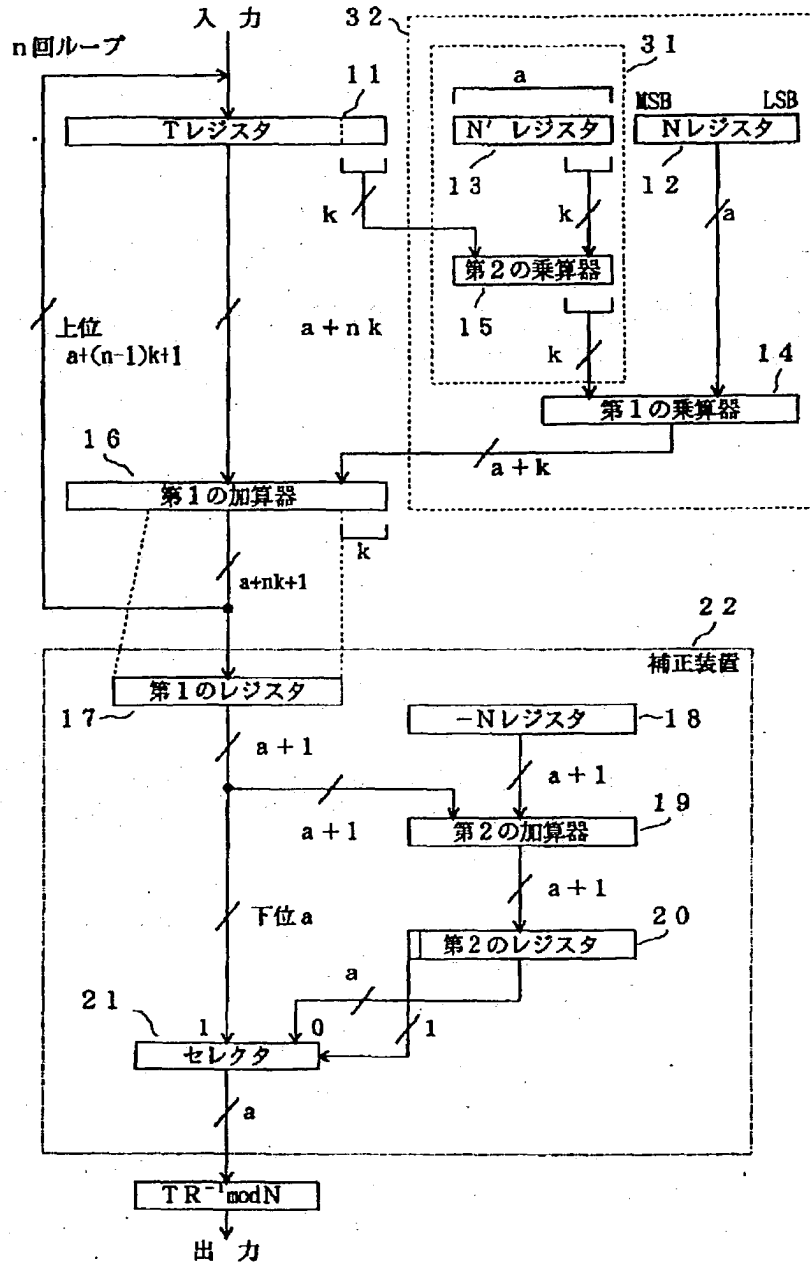


$$AB \rightarrow ABR^{-1} \bmod N \quad R = b^* \quad b = 2^k$$



【図8】

モンゴメリのアルゴリズムを用いた一般的な剰余計算装置の構成図



フロントページの続き

(72)発明者 秋山 良太  
 神奈川県川崎市中原区上小田中1015番地  
 富士通株式会社内